

## Burs 2.

### clientul

IP server → Port  
 $c = \text{SOCKET}()$   
 $\text{connect}(c, \{ -\text{IP\_server} \\ -\text{port} \})$

$\text{send}(c,$   
 $\text{recv}(c,$   
 $\text{close}(c)$

### serverul

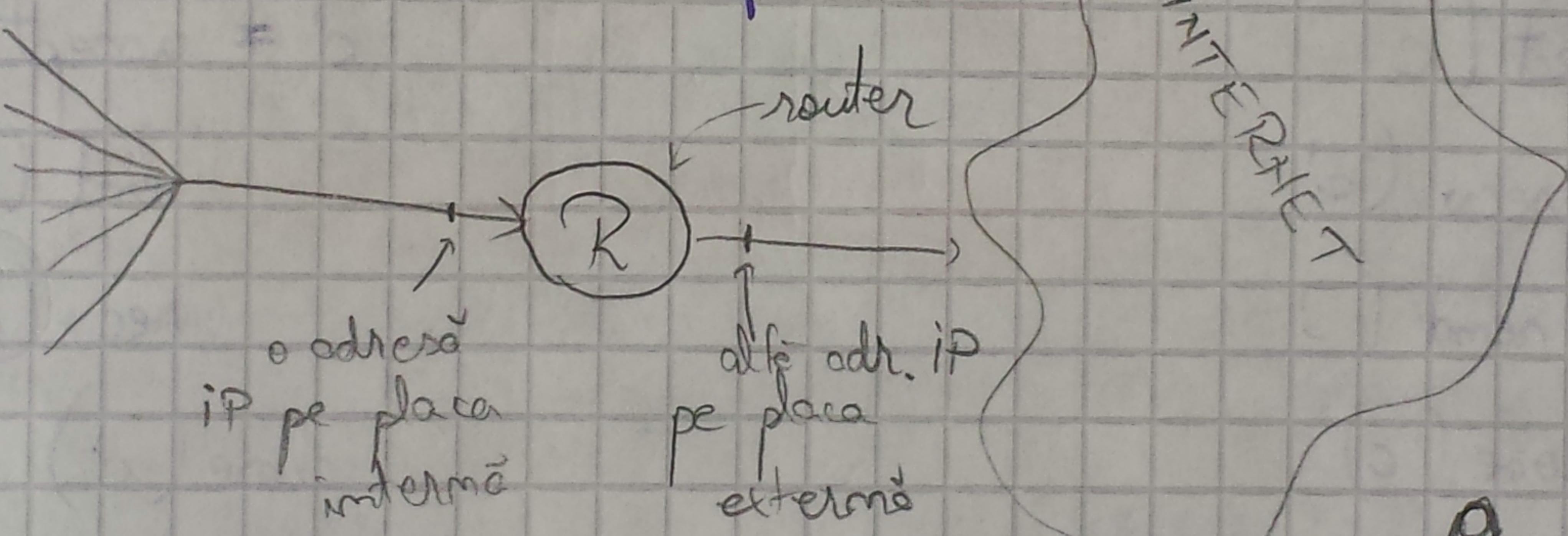
$s = \text{SOCKET}()$   
 $\text{bind}(s, \{ -\text{port} \\ -\text{ADDR\_ANY} \})$

$\text{listen}(s)$   
 $c' = \text{accept}(s, \{ -\text{IP\_client} \\ -\text{port\_client} \})$

//  $c' \rightarrow$  descriptor de socket

$\text{recv}(c'$   
 $\text{send}(c'$   
 $\text{close}(c')$

$\text{close}(s)$



! 2 procese ~~sunt~~ diferite pe port

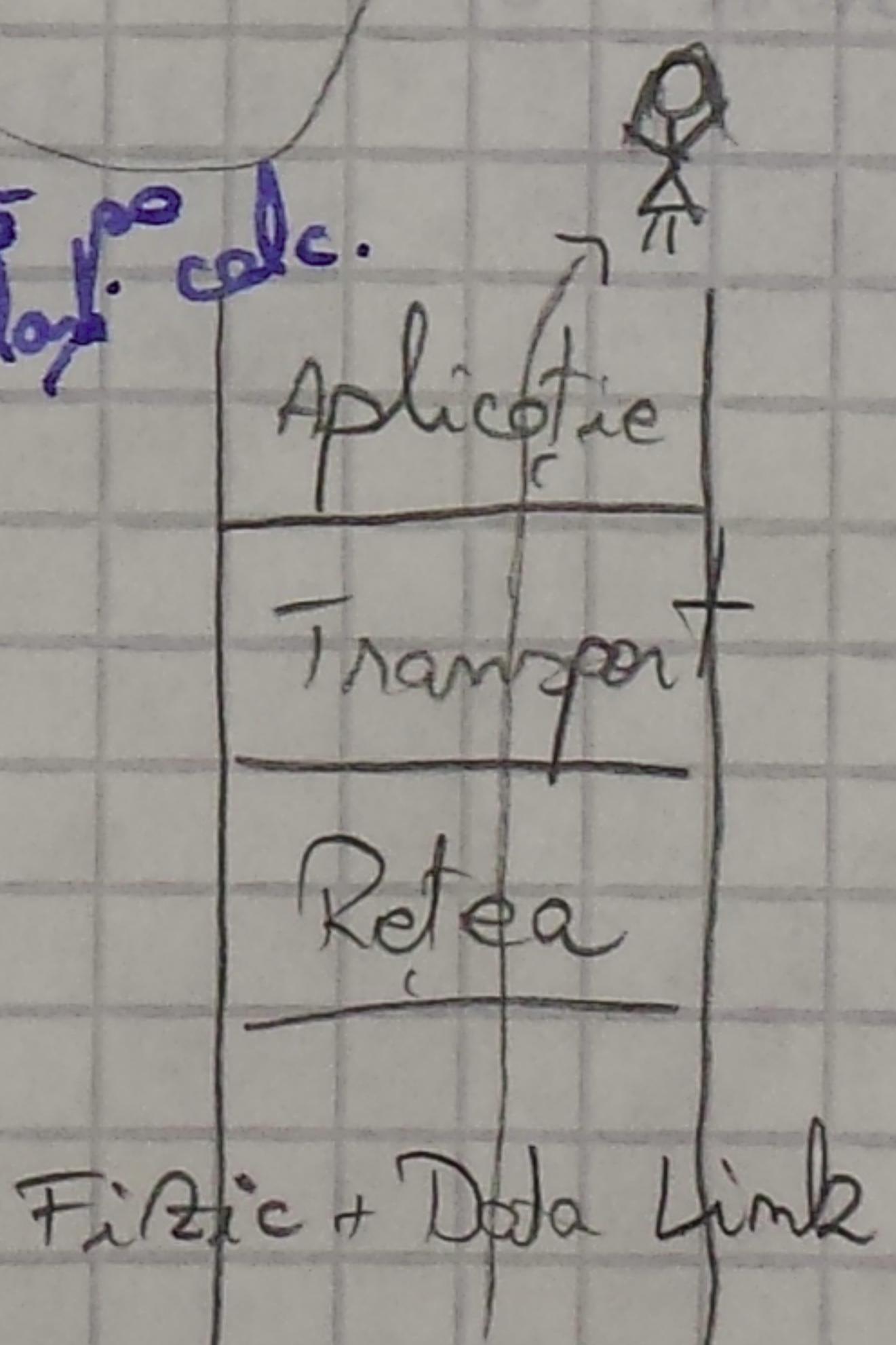
~~accesă date de pe acelasi port că~~  
~~server-ului?~~

acțepta pe același port ?

Da, dacă : ① ~~nu~~ nu acțeaptă

ambide pe ADDR\_ANY

② unul este TCP / IP, altul UDP



bind-ul nu este atât de important la client deoarece portul clientului este ales de către sistemul de operare  
dintre porturile libere

- send ( c, de unde scriu , cât scriu )
- recv ( c, unde citesc , cât citesc )

items

html

\* mthds

mthds

### TEMĂ DE CASĂ:

- să se implementeze mthds în C

0.0.0.0 → ADDR\_ANY

13.X.2015

### Curs 3. Server TCP concurent.

client

server

```
connect (c
    recv (c'
    send (c'
    close (c)
```

```
c' = accept ( s
    send (c'
```

```
    recv (c'
```

```
    close (c')
```

while

```
c' = accept ( s
```

```
if ( fork () == 0) {
```

```
deservire ( c'); } don (c')
```

```
} exit (0);
```